Fiche TD5 : Grammaires algébriques

Exercice 1 (*) *Du langage à la grammaire*

Dans chacun des cas suivants, on demande d'écrire une grammaire engendrant le langage proposé.

1. $L_1 = a^\star bc^\star$.
2. L_2 est le langage sur $\{a, b\}$ des mots ayant autant de a que de b.
3. L_3 est le langage sur $\{a, =, +\}$ des mots représentant une addition correcte de deux suites de caractères a. Par exemple, $aa + aaa = aaaaa \in L_3$ mais $a + a = a$ ou $a + a + a = aaa$ n'en font pas partie.
4. L_4 est le langage dont les mots sont les listes de chiffres bien formées en Ocaml. Par exemple, $[\,], [0; 1]$ et $[5; 4; 1; 1; 2]$ sont des mots de L_4 mais $[10; 1]$, $[1;]$ ou $[-1; 5]$ n'en font pas partie.
5. L_5 est le langage dont les mots sont les objets que l'on peut construire à l'aide du type récursif suivant :

   ```
   type arbre_binaire = Vide | Feuille | Noeud of arbre_binaire * arbre_binaire.
   ```

   Par exemple, Vide et Noeud(Noeud(Feuille, Vide), Feuille) sont des mots de L_5 alors que Noeud(Noeud,Noeud) ou (Noeud(Feuille, Vide)) n'en font pas partie.
6. L_6 est le langage des formules du calcul propositionnel sur l'ensemble de variables $\{p, q, r\}$.

Exercice 2 (*) *De la grammaire au langage*

1. On considère la grammaire G_1 d'axiome S et dont les règles sont $S \to aSa \,|\, bSb \,|\, \varepsilon$. Montrer que $L(G_1)$ est exactement l'ensemble des palindromes de longueur paire sur $\{a, b\}$.
2. On considère la grammaire G_2 d'axiome S et dont les règles sont $S \to aSS \,|\, b$. Montrer que $L(G_2)$ est l'ensemble des mots u tels que $|u|_a = |u|_b - 1$ et pour tout préfixe strict v de u (c'est-à-dire, $v \neq u$), $|v|_b \leq |v|_a$. *Indication : S'inspirer de la question 3 de l'exercice 3 du TD2.*

Culture générale : Le langage de la deuxième question est le langage de Łukasiewicz, généralement noté Ł. C'est le langage des expressions préfixes à un opérateur bianire (ici noté a). Il entretient des rapports étroits avec le langage de Dyck.

Exercice 3 (**) *Grammaires et langues naturelles*

On considère deux grammaires dont l'axiome est P dans les deux cas : G_1 permet de produire quelques phrases en anglais et G_2 fait de même en français. Les règles de ces grammaires sont données par les règles suivantes (les majuscules correspondent aux non terminaux et les minuscules aux terminaux) :

Règles de G_1

$P \to NV$
$N \to NP$
$N \to$ the dog | the stick | the fire
$V \to$ burned | bit | beat

Règles de G_2

$P \to SVC$
$S \to N \,|\, NC$
$C \to$ avec $N \,|\, \varepsilon$
$V \to WN$
$N \to$ elle | une femme | un téléscope
$W \to$ voit

1. Donner un arbre syntaxique pour le mot suivant engendré par la grammaire G_1 :

   the dog the stick the fire burned beat bit

   Comment traduirait-on cette phrase en français ?
2. On considère le mot suivant engendré par la grammaire G_2 :

   elle voit une femme avec un téléscope

   Donner deux sens possibles à cette phrase. Montrer que G_2 est une grammaire ambiguë.

Remarque : De manière générale, les grammaires algébriques ne sont pas tout à fait adaptées à la description d'une langue naturelle. Chomsky lui-même convenait de cette limite et proposa pour la lever la notion de grammaire transformationnelle. Aujourd'hui, cette notion est généralement remplacée par celle de grammaire faiblement contextuelle.

Exercice 4 (**) *Ambiguité et langages rationnels*

1. On considère la grammaire G d'axiome S et dont les règles sont

$$S \to abA \,|\, AbA \text{ et } A \to \varepsilon \,|\, aA$$

   a) Quel est le langage engendré par G ?

   b) Montrer que G est ambiguë.

   c) Exhiber une grammaire non ambiguë qui engendre le langage $L(G)$.

2. Montrer de manière générale qu'un langage rationnel n'est jammais inhéremment ambigu, autrement dit, montrer que pour tout langage rationnel, il existe une grammaire non ambiguë qui l'engendre.

Exercice 5 (**) *Désambiguïsation*

1. On cherche à construire une grammaire destinée à écrire des expressions arithmétiques construites à partir de l'opérateur binaire de soustraction et dont les opérandes sont des entiers qu'on notera génériquement e (autrement dit, e peut être dérivé en n'importe quel entier). On propose les règles suivantes pour une telle grammaire : $S \to S - S \,|\, e$.

   a) A l'aide du mot $m = 10 - 2 - 3$, montrer que cette grammaire est ambiguë. Pourquoi est-ce gênant ?

   b) On modifie alors les règles de la grammaire comme suit :

$$\begin{aligned} S &\to S - T \,|\, T \\ T &\to e \end{aligned}$$

   Dessiner l'arbre syntaxique du mot m. La grammaire obtenue est-elle ambiguë ?

2. Dans cette question, on considère une grammaire décrivant les instructions d'un langage de programmation dont l'axiome est S, les non terminaux sont S et I et dont les règles sont :

$$\begin{aligned} S &\to I \\ I &\to \text{if } e \text{ then } I \mid \text{if } e \text{ then } I \text{ else } I \mid a \end{aligned}$$

   a) Montrer que cette grammaire est ambiguë.

   b) Proposer une solution pour rendre cette grammaire non ambiguë. Est-il possible de rendre cette grammaire non ambiguë sans modifier le langage décrit ?

Exercice 6 (**) *Théorème de lecture unique*

Le but de cet exercice est de montrer le théorème de lecture unique :

> Soit F une formule du calcul propositionnel sur un ensemble de variables V. Alors, on est dans un et un seul des cas suivants :
>
> - F est égale à un et un seul des éléments de $V \cup \{\top, \bot\}$,
> - Il existe une unique formule G telle que $F = \neg G$,
> - Il existe un unique couple de formules (G, H) et un unique symbole $\alpha \in \{\vee, \wedge \Rightarrow, \Leftrightarrow\}$ tel que $F = (G \alpha H)$

Considérons l'alphabet $\Sigma = V \cup \{(,), \neg, \wedge, \vee, \Rightarrow, \Leftrightarrow\}$ et notons $\mathcal{F}$ l'ensemble des formules du calcul propositionnel sur l'ensemble de variable V. Remarquons qu'un élément de $\mathcal{F}$ n'est rien d'autre qu'un mot de $\Sigma^\star$.

1. Montrer que pour toute formule F, on a $|F|_(= |F|_)$.

2. Montrer que pour tout préfixe u d'une formule F, $|u|_(\geq |u|_)$ et que l'inégalité est stricte si le premier symbole de F est (et que u est propre (non égal à F ni à ε).

3. Montrer qu'un préfixe propre d'une formule n'est pas une formule.

4. Montrer le théorème de lecture unique.

5. Le langage des formules du calcul propositionnel sur V est-il (intrinsèquement) ambigu ?

Remarque : On comprend mieux pourquoi on peut se permettre de parler de l'arbre syntaxique d'une formule du calcul propositionnel plutôt que d'UN arbre syntaxique : on vient de montrer qu'il n'y en a qu'un. Remarquez aussi que le vocabulaire est bien fait : l'arbre syntaxique d'une formule est un arbre syntaxique.

Exercice 7 (***) *Lemme d'Ogden et applications*

On admet dans cet exercice le lemme d'Ogden :

> Soit $G = (\Sigma, V, S, \mathcal{R})$ une grammaire algébrique et $X \in V$. Alors il existe un entier N tel que tout mot $m \in L(G, X)$ ayant au moins N lettres marquées se factorise en $m = xuyvz$ avec
>
> 1. $X \Longrightarrow^* xAz$, $A \Longrightarrow^* uAv$ et $A \Longrightarrow^* y$ avec $A \in V$
> 2. (x, u et y contiennent des lettres marquées) ou (y, v et z contiennent des lettres marquées).
> 3. uyv contient moins de N lettres marquées.

1. A l'aide du lemme d'Ogden, prouver le lemme d'itération vu en cours.

2. Considérons le langage $L = \{a^n b^n c^m \mid n, m \in \mathbb{N}\} \cup \{a^n b^m c^m \mid n, m \in \mathbb{N}\}$.

   a) Montrer que L est algébrique.

   b) A l'aide du lemme d'Ogden, montrer que L est intrinsèquement ambigu. *Indication : Considérer le mot $a^N b^N c^{N+N!}$ avec N donné par le lemme d'Ogden et où tous les b sont marqués et montrer que $a^{N+N!} b^{N+N!} c^{N+N!}$ admet deux arbres de dérivation différents.*

Remarques : Parfois, le terme "lemme d'itération" réfère au lemme d'Ogden et non à la version simplifiée que j'ai présentée en cours. La démonstration du lemme d'Ogden est non triviale, vous pourrez la trouver à la page 100 de Langages formels, calculabilité et complexité par Olivier Carton.